1. Write a java programme to sort the integers 8, 4, 3, 5, 6 and the alphabetical string C, O, I, P, U, in ascending order. Show the resulting output.

CODE:

Package Integers;

import java.util.Arrays;

public class SortIntegersAndStrings {

public static void main(String[] args) {

//integer array

Integer[] integers = { 8, 4, 3, 5, 6 };

//string array

String[] strings = { "C", "O", "I", "P", "U" };

//sort the integer array

Arrays.sort(integers);

//sort the string array

Arrays.sort(strings);

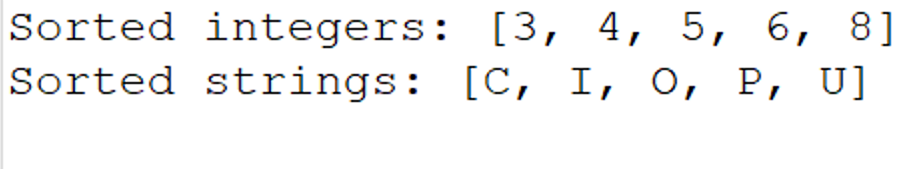
//print the sorted arrays

System.out.println("Sorted integers: " + Arrays.toString(integers));

System.out.println("Sorted strings: " + Arrays.toString(strings));

}

}  
  
OUTPUT:



1. Write a Java program to implement the bubble sort algorithm to sort an array of integers in ascending order.

CODE:

package Bubble;

public class Bubble\_Sorting {

//method to perform bubble sort on an array

public static void bubbleSort(int[] array) {

int n = array.length;

boolean swapped;

//loop over each element in the array

for (int i = 0; i < n - 1; i++) {

swapped = false;

//inner loop for comparing adjacent elements

for (int j = 0; j < n - 1 - i; j++) {

//swap if the element is greater than the next element

if (array[j] > array[j + 1]) {

int temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

swapped = true;

}

}

if (!swapped) {

break;

}

}

}

//main method to test the bubble sort algorithm

public static void main(String[] args) {

int[] array = {64, 34, 25, 12, 22, 11, 90};

System.out.println("Original array:");

for (int num : array) {

System.out.print(num + " ");

}

System.out.println();

bubbleSort(array);

System.out.println("Sorted array:");

for (int num : array) {

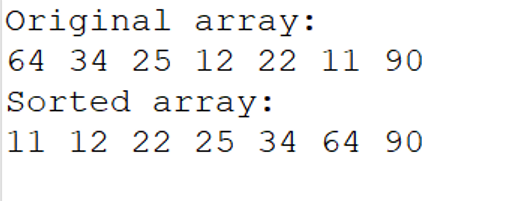
System.out.print(num + " ");

}

}

}

OUTPUT:



1. Write a program to input an array 10 elements and print the cube of prime numbers in it.

CODE:

package Array;

import java.util.Scanner;

public class PrimeCubes {

//method to check if a number is prime

public static boolean isPrime(int num) {

if (num <= 1) {

return false;

}

for (int i = 2; i <= Math.sqrt(num); i++) {

if (num % i == 0) {

return false;

}

}

return true;

}

//method to calculate the cube of a number

public static int cube(int num) {

return num \* num \* num;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int[] array = new int[10];

//input 10 elements into the array

System.out.println("Enter 10 integers:");

for (int i = 0; i < 10; i++) {

array[i] = scanner.nextInt();

}

//print the cube of prime numbers in the array

System.out.println("Cubes of prime numbers in the array:");

for (int num : array) {

if (isPrime(num)) {

System.out.println("Cube of " + num + " is " + cube(num));

}

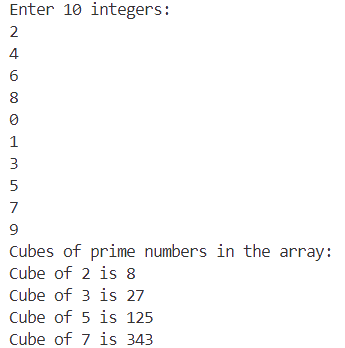
}

scanner.close();

}

}

OUTPUT:



1. Write a java program to implement integer wrapper class methods. (Any 5 methods)

CODE:

package Wrapper;

public class IntegerWrapperClassMethods {

public static void main(String[] args) {

//using the Integer.valueOf() method

Integer intObj = Integer.valueOf(123);

System.out.println("Integer.valueOf(123): " + intObj);

//using the Integer.parseInt() method

int intValue = Integer.parseInt("456");

System.out.println("Integer.parseInt(\"456\"): " + intValue);

//using the Integer.toString() method

String intString = Integer.toString(789);

System.out.println("Integer.toString(789): " + intString);

//using the Integer.compare() method

int compareResult = Integer.compare(10, 20);

System.out.println("Integer.compare(10, 20): " + compareResult); // Output will be negative because 10 < 20

//using the Integer.equals() method

Integer intObj1 = Integer.valueOf(100);

Integer intObj2 = Integer.valueOf(100);

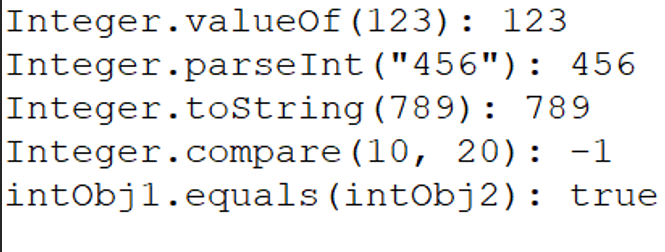
boolean isEqual = intObj1.equals(intObj2);

System.out.println("intObj1.equals(intObj2): " + isEqual); // Output will be true because 100 equals 100

}

}

OUTPUT:



1. Write a java program to implement double wrapper class methods. (Any 5 methods)

CODE:

package Double;

public class DoubleWrapperClassMethods {

public static void main(String[] args) {

//using the Double.valueOf() method

Double doubleObj = Double.valueOf(123.45);

System.out.println("Double.valueOf(123.45): " + doubleObj);

//using the Double.parseDouble() method

double doubleValue = Double.parseDouble("456.78");

System.out.println("Double.parseDouble(\"456.78\"): " + doubleValue);

//using the Double.toString() method

String doubleString = Double.toString(789.01);

System.out.println("Double.toString(789.01): " + doubleString);

//using the Double.compare() method

int compareResult = Double.compare(10.5, 20.5);

System.out.println("Double.compare(10.5, 20.5): " + compareResult); // Output will be negative because 10.5 < 20.5

//using the Double.isNaN() method

Double NaNValue = Double.NaN;

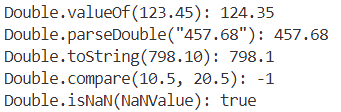
boolean isNaN = Double.isNaN(NaNValue);

System.out.println("Double.isNaN(NaNValue): " + isNaN); // Output will be true because NaNValue is Not-a-Number

}

}

OUTPUT:



1. Write a java program to implement float wrapper class methods. (Any 5 methods)

CODE:

package Implement;

public class FloatWrapperClassMethods {

    public static void main(String[] args) {

        //using the Float.valueOf() method

        Float floatObj = Float.valueOf(124.35f);

        System.out.println("Float.valueOf(124.35f): " + floatObj);

        //using the Float.parseFloat() method

        float floatValue = Float.parseFloat("457.68");

        System.out.println("Float.parseFloat(\"457.68\"): " + floatValue);

        //using the Float.toString() method

        String floatString = Float.toString(798.10f);

        System.out.println("Float.toString(798.10f): " + floatString);

        //using the Float.compare() method

        int compareResult = Float.compare(5.6f, 10.5f);

        System.out.println("Float.compare(5.6f, 10.5f): " + compareResult); //output will be negative

        //using the Float.isNaN() method

        Float NaNValue = Float.NaN;

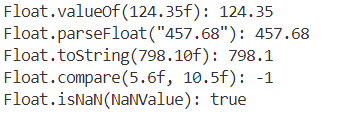
        boolean isNaN = Float.isNaN(NaNValue);

        System.out.println("Float.isNaN(NaNValue): " + isNaN); //output will be true because NaNValue is Not-a-Number

    }

}

OUTPUT:



1. Write a Java program to validate email addresses using regular expressions. The email should have the format username@domain.com where username and domain can contain alphanumeric characters, dots, and hyphens.

CODE:

package Email;

import java.util.Scanner;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class EmailValidator {

//method to validate email addresses

public static boolean isValidEmail(String email) {

//regular expression for validating an email address

String emailRegex = "^[a-zA-Z0-9.\_-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,6}$";

//compile the regular expression into a pattern

Pattern pattern = Pattern.compile(emailRegex);

//match the input email address against the pattern

Matcher matcher = pattern.matcher(email);

//return true if the email matches the pattern, false otherwise

return matcher.matches();

}

public static void main(String[] args) {

//create a Scanner object to read input

Scanner s = new Scanner(System.in)

System.out.println("Enter an email address to validate:");

String email = s.nextLine();

//validate the entered email address and print the result

if (isValidEmail(email)) {

System.out.println(email + " is a valid email address");

} else {

System.out.println(email + " is not a valid email address.");

}

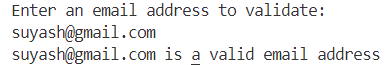
s.close();

}

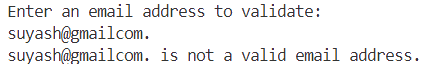
}

OUTPUT:

RIGHT OUTPUT:



WRONG OUTPUT:



1. Create a Java program to validate phone numbers. The format should be (xxx) xxx-xxxx where x is a digit.

CODE:

package Phone;

import java.util.Scanner;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class PhoneNumberValidator {

    //method to validate phone numbers

    public static boolean isValidPhoneNumber(String phoneNumber) {

        //regular expression for validating a phone number

        String phoneNumberRegex = "^\\(\\d{3}\\) \\d{3}-\\d{4}$";

        //compile the regular expression into a pattern

        Pattern pattern = Pattern.compile(phoneNumberRegex);

        //match the input phone number against the pattern

        Matcher matcher = pattern.matcher(phoneNumber);

        //return true if the phone number matches the pattern, false otherwise

        return matcher.matches();

    }

    public static void main(String[] args) {

        //create a Scanner object to read input

        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter a phone number to validate (format: (xxx) xxx-xxxx):");

        String phoneNumber = scanner.nextLine();

        //validate the phone number

        if (isValidPhoneNumber(phoneNumber)) {

            System.out.println(phoneNumber + " is a valid phone number.");

        } else {

            System.out.println(phoneNumber + " is not a valid phone number.");

        }

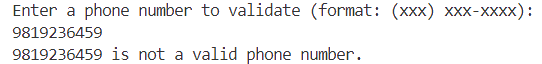
        scanner.close();

    }

}

OUTPUT:

WRONG OUTPUT:



RIGHT OUTPUT:

